UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/886,585	06/21/2001	Daniel M. Lavery	2207/11237	6346

59796 7590 06/28/2007

INTEL CORPORATION
c/o INTELLEVATE, LLC
P.O. BOX 52050
MINNEAPOLIS, MN 55402

EXAMINER	
RAMPURIA, SATISH	
ART UNIT	**PAPER NUMBER**
2191	
MAIL DATE	**DELIVERY MODE**
06/28/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	09/886,585	LAVERY ET AL.	
	Examiner	Art Unit	
	Satish S. Rampuria	2191	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *20 February 2007*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-30* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-30* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

Response to Amendment

1. This action is in response to the appeal brief received on 02/20/2007.

2. Claims 1-30 are pending.

3. In view of the appeal brief filed on 02/20/2007, PROSECUTION IS HEREBY REOPENED. New grounds of rejection set forth below.

To avoid abandonment of the application, appellant must exercise one of the following two options:

(1) file a reply under 37 CFR 1.111 (if this Office action is non-final) or a reply under 37 CFR 1.113 (if this Office action is final); or,

(2) initiate a new appeal by filing a notice of appeal under 37 CFR 41.31 followed by an appeal brief under 37 CFR 41.37. The previously paid notice of appeal fee and appeal brief fee can be applied to the new appeal. If, however, the appeal fees set forth in 37 CFR 41.20 have been increased since they were previously paid, then appellant must pay the difference between the increased fees and the amount previously paid.

A Supervisory Patent Examiner (SPE) has approved of reopening prosecution by signing below.

Response to Arguments

4. Applicant's arguments with respect to claims have been considered but are moot in view of the new ground(s) of rejection.

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. Claim 1-30 are rejected under 35 U.S.C. 103(a) as being unpatentable over US Patent No. 6,389,446 to Torii, hereinafter called Torii, in view of US Patent No. 5,978,578 to Azarya et al., hereinafter called Azarya.

Per claim 1:

Torii discloses:

- A method for executing a code (col. 2, lines 56-57 "executes a plurality of threads of instruction streams"), comprising:
- receiving an instruction (col. 6, lines 36-37 "thread manager 5 receives the request");
- determining whether the instruction is a trigger instruction (col. 6, lines 36-42 "determines whether the thread being executed by the one of thread processors... It is determined by checking the content of child thread processor entry number 12 in thread status table 9");
- executing an auxiliary code, the auxiliary code executing separate from the instruction (col. 4, lines 32-33 "a thread generation instruction 2 generates thread #1 (1b) from thread #0 (1a)"). As understood from the specification (paragraph [0012-0015]) the auxiliary code nothing but executing in parent child environment.

Torii does not explicitly disclose selecting an entry in a trigger table, if the instruction is a trigger instruction, the entry associated with the trigger instruction and entry is referenced by the trigger table.

However, Azarya in an analogous computer system discloses selecting an entry in a trigger table (col. 5, lines 6-7 "generating an event trigger table"), if the instruction is a trigger instruction, the entry associated with the trigger instruction and entry is referenced by the trigger table (col. 5, lines 7- "the event trigger table comprising a plurality of event trigger entries, each event trigger entry corresponding to an action that references the particular external input signal or entity that points thereto, generating a plurality of actions, each of the actions comprising at least one frame, the actions, the actions representing the generation of output signals and/or the modification of the internal entities, and wherein the plurality of pointer tables, the event trigger table and the plurality of actions generated in accordance with the event triggers").

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the method of selecting an entry in a trigger table, the entry associated with the trigger instruction and entry is referenced by the trigger table as taught by Azarya into the method of executing the code as taught by Torii. The modification would be obvious because of one of ordinary skill in the art would be motivated to select an entry in a trigger table to provide an automated system that will reduce low life cycle costs, maximized machine uptime, minimized machine downtime and optimize the performance as suggested by Azarya (col. 1, lines 34-50).

Per claim 2:

The rejection of claim 1 is incorporated, and further, Torii disclose:

- spawning a new thread, the new thread executing instructions included in the auxiliary code (col. 4, lines 44-45 "the parent thread of thread #1 (1b) is thread

#0 (1a) while the child thread of thread #1 (1b) is thread #2 (1c)"). As understood from the specification (paragraph [0012-0015]) the auxiliary code nothing but executing in parent child environment

Per claim 3:

The rejection of claim 2 is incorporated, and further, Torii disclose:

- executing the new thread concurrently with a parent thread, the parent thread including the trigger instruction (col. 4, lines 44-45 "the parent thread of thread #1 (1b) is thread #0 (1a) while the child thread of thread #1 (1b) is thread #2 (1c)").

Per claim 4:

Torii discloses:

- A computer-implemented method for executing a code (col. 2, lines 56-57 "executes a plurality of threads of instruction streams"), comprising:
- receiving a trigger instruction (col. 6, lines 36-37 "thread manager 5 receives the request");
- executing a p-slice code (col. 4, lines 32-33 "a thread generation instruction 2 generates thread #1 (1b) from thread #0 (1a)"). As understood from the specification (See paragraph [0012-0015]) the auxiliary code nothing but executing in parent child environment.

Torii does not explicitly disclose selecting an entry in a trigger table, the entry associated with the trigger instruction and entry is referenced by the trigger table.

However, Azarya in an analogous computer system discloses selecting an entry in a trigger table (col. 5, lines 6-7 "generating an event trigger table"), the entry associated with the trigger instruction and entry is referenced by the trigger table (col. 5, lines 7- "the event trigger table comprising a plurality of event trigger entries, each event trigger entry corresponding to an action that references the particular external input signal or entity that points thereto, generating a plurality of actions, each of the actions comprising at least one frame, the actions, the actions representing the generation of output signals and/or the modification of the internal entities, and wherein the plurality of pointer tables, the event trigger table and the plurality of actions generated in accordance with the event triggers")..

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the method of selecting an entry in a trigger table, the entry associated with the trigger instruction and entry is referenced by the trigger table as taught by Azarya in the method of executing the code as taught by Torii. The modification would be obvious because of one of ordinary skill in the art would be motivated to select an entry in a trigger table to provide an automated system that will reduce low life cycle costs, maximized machine uptime, minimized machine downtime and optimize the performance as suggested by Azarya (col. 1, lines 34-50).

Per claim 5:

The rejection of claim 4 is incorporated, and further, Torii disclose:

- spawning a new thread, the new thread executing instructions included in the auxiliary code (col. 4, lines 44-45 "the parent thread of thread #1 (1b) is thread #0 (1a) while the child thread of thread #1 (1b) is thread #2 (1c)"). As understood from the specification (paragraph [0012-0015]) the auxiliary code nothing but executing in parent child environment.

Per claim 6:

The rejection of claim 5 is incorporated, and further, Torii disclose:

- executing the new thread concurrently with a parent thread, the parent thread including the trigger (col. 4, lines 44-45 "the parent thread of thread #1 (1b) is thread #0 (1a) while the child thread of thread #1 (1b) is thread #2 (1c)").

Per claim 7:

The rejection of claim 6 is incorporated, and further, Torii disclose:

- storing state information from the parent thread before spawning the new thread (col. 11, lines 21-22* "Therefore, thread manager 28 (or 32) stores the child thread start information into thread saving buffer 31 (or 37)").

Per claim 8:

The rejection of claims 7 and 9 are incorporated, respectively, and further, Torii disclose:

- copying the state information for use in the new thread (col. 11, lines 6-11 "Thread information saving line 37 saves a thread execution start address and data essential for starting a child thread's execution...").

Per claim 9:

The rejection of claim 6 is incorporated, and further, Torii disclose:

- storing a register value of the parent thread before spawning the new thread (col. 11, lines 6-11 "Thread information saving line 37 saves a thread execution start address and data essential for starting a child thread's execution...").

Per claim 10:

The rejection of claim 9 is incorporated, and further, Torii disclose:

- copying the register value of the parent thread for use in the new thread (col. 11, lines 6-11 "Thread information saving line 37 saves a thread execution start address and data essential for starting a child thread's execution...").

Per claim 11:

The rejection of claim 4 is incorporated, and further:

- wherein the entry in the trigger table is selected by associative lookup of the trigger instruction. The recited in this claim are similar to those recited in claim 4 and rejected under the same rational set forth in connection with the rejection of claim 4 above.

Per claim 12:

The rejection of claim 4 is incorporated, and further, Torii disclose:

- reading an instruction pointer for the p-slice code from the entry in the trigger table (See FIG. 4, element 9 and related discussion).

Claim 13 is the computer program product claim corresponding to method claim 1 and rejected under the same rational set forth in connection with the rejection of claim 1 above.

Claim 14 is the computer program product claim corresponding to method claim 2 and rejected under the same rational set forth in connection with the rejection of claim 2 above.

Claims 15, 18 and 21 are the system claim corresponding to method claim 1 and rejected under the same rational set forth in connection with the rejection of claim 1 above.

Claims 16 and 19 are the system claim corresponding to method claim 2 and rejected under the same rational set forth in connection with the rejection of claim 2 above.

Claims 17 and 20 are the system claim corresponding to method claim 7 and rejected under the same rational set forth in connection with the rejection of claim 7 above.

Per claims 22 and 24:

Torii discloses:

- A computer-implemented method for compiling (col. 2, lines 56-57 "executes a plurality of threads of instruction streams"), comprising:
- receiving a function body, the function body comprising a trigger instruction (col. 6, lines 36-37 "thread manager 5 receives the request");
- outputting an auxiliary code associated with the function body and the trigger instruction (col. 4, lines 32-33 "a thread generation instruction 2 generates thread #1 (1b) from thread #0 (1a)"). As understood from the specification (See paragraph [0012-0015]) the auxiliary code nothing but executing in parent child environment.

Torii does not explicitly disclose creating an entry in a trigger table the entry associated with the trigger instruction and the auxiliary code.

However, Azarya, in an analogous computer system discloses creating an entry in a trigger table (col. 5, lines 6-7 "generating an event trigger table") the entry associated with the trigger instruction and the auxiliary code (col. 5, lines 7- "the event trigger table comprising a plurality of event trigger entries, each event trigger entry corresponding to an action that references the particular external input signal or entity that points thereto, generating a plurality of actions, each of the actions comprising at least one frame, the actions, the actions representing the generation of output signals and/or the modification of the internal entities, and wherein the plurality of pointer tables, the event trigger table and the plurality of actions generated in accordance with the event triggers").

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the method of creating an entry in a

trigger table as taught by Azarya in the method of executing the code as taught by Torii. The modification would be obvious because of one of ordinary skill in the art would be motivated to select an entry in a trigger table to provide an automated system that will reduce low life cycle costs, maximized machine uptime, minimized machine downtime and optimize the performance as suggested by Azarya (col. 1, lines 34-50).

Per claim 23:

The rejection of claims 22 and 24 is incorporated, respectively, and further, Torii disclose:

- creating a stub block, the stub block comprising a spawn instruction, the spawn instruction configured to spawn a new thread, the new thread configured to execute the auxiliary code (col. 4, lines 44-45 "the parent thread of thread #1 (1b) is thread #0 (1a) while the child thread of thread #1 (1b) is thread #2 (1c)"). It is inherent to create the stub block in order to execute the spawn instructions.

Per claim 24:

Torii discloses:

- A method for compiling (col. 2, lines 56-57 "executes a plurality of threads of instruction streams"), comprising:
- receiving a function body, the function body comprising a trigger instruction (col. 6, lines 36-37 "thread manager 5 receives the request");

- outputting a p-slice code associated with the function body and the trigger instruction (col. 4, lines 32-33 "a thread generation instruction 2 generates thread #1 (1b) from thread #0 (1a)"). As understood from the specification (See paragraph [0012-0015]) the auxiliary code nothing but executing in parent child environment.

Torii does not explicitly disclose creating an entry in a trigger table, the entry associated with the trigger instruction and the p-slice code.

However, Azarya, in an analogous computer system discloses creating an entry in a trigger table (col. 5, lines 6-7 "generating an event trigger table"), the entry associated with the trigger instruction and the p-slice code (col. 5, lines 7- "the event trigger table comprising a plurality of event trigger entries, each event trigger entry corresponding to an action that references the particular external input signal or entity that points thereto, generating a plurality of actions, each of the actions comprising at least one frame, the actions, the actions representing the generation of output signals and/or the modification of the internal entities, and wherein the plurality of pointer tables, the event trigger table and the plurality of actions generated in accordance with the event triggers").

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the method of creating an entry in a trigger table as taught by Azarya in the method of executing the code as taught by Torii. The modification would be obvious because of one of ordinary skill in the art would be motivated to select an entry in a trigger table to provide an automated system

that will reduce low life cycle costs, maximized machine uptime, minimized machine downtime and optimize the performance as suggested by Azarya (col. 1, lines 34-50).

Per claim 25:

The rejection of claim 24 is incorporated, respectively, and further, Torii disclose:

- receiving the p-slice code associated with the function body and the trigger instruction (col. 6, lines 36-37 "thread manager 5 receives the request").

Per claim 26:

The rejection of claim 24 is incorporated, respectively, and further, Torii disclose:

- generating the p-slice code associated with the function body and the trigger instruction (col. 4, lines 32-33 "a thread generation instruction 2 generates thread #1 (1b) from thread #0 (1a)"). As understood from the specification (See paragraph [0012-0015]) the auxiliary code nothing but executing in parent child environment.

Per claim 27:

The rejection of claim 24 is incorporated, respectively, and further:

- creating a stub block, the stub block comprising a spawn instruction, the spawn instruction configured to spawn a new thread, the new thread configured to execute the p-slice code. The recited in this claim are similar to those recited in

claim 23 and rejected under the same rational set forth in connection with the rejection of claim 23 above.

Per claim 28:

The rejection of claim 24 is incorporated, respectively, and further, Torii disclose:

- adding store instructions to the stub block, the store instructions configured to store state information of a current (col. 11, lines 6-11 "Thread information saving line 37 saves a thread execution start address and data essential for starting a child thread's execution...") the state information of the current thread including values contained in live-in registers of the new thread (col. 11, lines 24-29 "When thread processor... execution of thread #0, it enters a "free"-state... thread manager 28 (or 32) pulls the child thread start information out of thread saving buffer 31 (or 37), and uploads it into thread processor #0 to start the child thread, or thread #4").

Claims 29 and 30 are the computer program product claim corresponding to method claim 24 and rejected under the same rational set forth in connection with the rejection of claim 24 above.

Conclusion

7. Any inquiry concerning this communication or earlier communications from the examiner should be directed to **Satish S. Rampuria** whose telephone number is **(571)**

272-3732. The examiner can normally be reached on **8:30 am to 5:00 pm** Monday to Friday except every other Friday and federal holidays. Any inquiry of a general nature or relating to the status of this application should be directed to the **TC 2100 Group receptionist: 571-272-2100**

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, **Wei Y. Zhen** can be reached on **(571) 272-3708**. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Satish S. Rampuria
Patent Examiner/Software Engineer
Art Unit 2191

WEI ZHEN
SUPERVISORY PATENT EXAMINER